



RPC BROKER SYSTEMS MANUAL

Version 1.1*9

January 2000

Department of Veterans Affairs
VHA CIO Technical Services

Preface

The *Remote Procedure Call (RPC) Broker Systems Manual* provides descriptive information and instructions on the use of the RPC Broker (also referred to as "Broker") software within the VA's Veterans Health Information Systems and Technology Architecture (**VISTA**) client/server environment. The emphasis is on the use of Borland's Delphi software. However, the RPC Broker does support other client environments.

This document is intended for the **VISTA** development community, the Information Resource Management (IRM) staff, and clinicians using Broker-based client/server applications. A wider audience of technical personnel engaged in operating and maintaining the Department of Veterans Affairs (VA) software may also find it useful as a reference.

Table of Contents

Orientation	Orientation-1
Introduction	Introduction-1
1. System Features.....	1-1
Client Features.....	1-1
RPC Broker Client Agent	1-1
"Connect To" Application Window	1-2
Edit Broker Servers Program.....	1-2
Stand-alone Programs and their Associated Help Files	1-4
HOSTS File	1-5
What Happened to the Client Manager?	1-9
What Happened to the VISTA.INI File?.....	1-9
Server Features	1-11
Menu for System Managers.....	1-11
Broker Listeners and Ports	1-11
Starting And Stopping Listeners	1-12
RPC BROKER SITE PARAMETERS File.....	1-14
Integrated Single Signon for Multiple User Sessions	1-15
RPC Broker Message Structure	1-18
Client/Server Timeouts.....	1-19
Load Balancing on Alpha Systems.....	1-20
MSM for NT 4.3.0 MSERVER Replaces RPC Broker Listener Process	1-21
2. Security.....	2-1
Security Features	2-1
Validation of Connection Request	2-1
Validation of Users.....	2-1
Validation of RPCs	2-7
Sample Security Procedures	2-8
Security Features Tasks Summary	2-8
3. Troubleshooting.....	3-1
Test the Broker Using the RPC Broker Diagnostic Program.....	3-1
Verify and Test the Network Connection.....	3-4
Signon Delays.....	3-5
Glossary.....	Glossary-1
Index	Index-1

Table of Contents

Orientation

HOW TO USE THIS MANUAL

Throughout this manual, advice and instructions are offered regarding the use of the RPC Broker and the functionality it provides for **VISTA** and commercial off-the-shelf (COTS) software products.



There are no special legal requirements involved in the use of the RPC Broker's Interface.

This manual uses several methods to highlight different aspects of the material:

- Descriptive text is presented in a proportional font (as this is).
- "Snapshots" of computer online displays (i.e., roll-and-scroll dialogs) and computer source code are shown in a non-proportional font and enclosed within a box. Also included are Graphical User Interface (GUI) application window images (i.e., Windows' dialog boxes or forms).


NOTE: Author's comments within these snapshots are displayed in italics or as "callout" boxes (callout boxes refer to labels or descriptions, usually enclosed within a box, which point to specific areas of a displayed image). User's responses to online prompts will be boldface.

- Object Pascal code uses a combination of upper- and lowercase characters. All Object Pascal reserved words are in boldface type.
- All uppercase is reserved for the representation of M code, variable names, or the formal name of options, field and file names, and security keys (e.g., the XUPROGMODE key).
- Various symbols are used throughout the documentation to alert the reader to special information. The following table gives a description of each of these symbols:

Symbol	Description
	Used to inform the reader of general information including references to additional reading material
	Used to caution the reader to take special notice of critical information

COMMONLY USED TERMS

The following is a list of terms and their descriptions that you may find helpful while reading the RPC Broker documentation:

Term	Description
Client	A single term used interchangeably to refer to a user, the workstation (i.e., PC), and the portion of the program that runs on the workstation.
Component	<p>A software object that contains data and code. A component may or may not be visible.</p> <p> <i>For a more detailed description, see the Borland Delphi for Windows User Guide.</i></p>
GUI	The Graphical User Interface application that is developed for the client workstation.
Host	The term Host is used interchangeably with the term Server.
Server	The computer where the data and the RPC Broker remote procedure calls reside.



Please see the Glossary for additional terms and definitions.

ASSUMPTIONS ABOUT THE READER

This manual is written with the assumption that the reader is familiar with the following:

- **VISTA** computing environment (e.g., Kernel Installation and Distribution System [KIDS])
- VA FileMan data structures and terminology
- Microsoft Windows
- M programming language

No attempt is made to explain how the overall **VISTA** programming system is integrated and maintained. Such methods and procedures are documented elsewhere. We suggest you look at the various VA home pages on the World Wide Web for a general orientation to **VISTA**. For example, check out the following web addresses:

<http://vista2.med.va.gov/>

This manual does provide, however, an explanation of the RPC Broker, describing how it can be used in a client/server environment.

Readers who wish to learn more about the RPC Broker should consult the following:

- *RPC Broker V. 1.1 Installation Guide*
- *RPC Broker V. 1.1 Release Notes*
- *RPC Broker V. 1.1 Technical Manual*
- *RPC Broker V. 1.1 Getting Started with the Broker Development Kit* (written for programmers)
- *Online RPC Broker Developer's Guide* (i.e., BROKER.HLP, designed for programmers)
- RPC Broker Home Page at the following web address:

<http://vista2.med.va.gov/broker/>

This site contains additional information and documentation (e.g., Frequently Asked Questions [FAQs]) available in Hypertext Markup Language (HTML).

Introduction

OVERVIEW

The RPC Broker is part of the infrastructure of **VISTA**. It establishes a common and consistent foundation for client/server applications being written as part of **VISTA**.

The RPC Broker is a bridge connecting the client application front-end on the workstation (e.g., Delphi GUI applications) to the M-based data and business rules on the server. It links one part of a program running on a workstation to its counterpart on the server. Therefore, the RPC Broker assists in opening the traditionally proprietary **VISTA** software to Commercial Off-the-Shelf (COTS) and Hybrid Open Systems Technology (HOST) products.

The RPC Broker includes the following:

- A common communications driver interface that handles the device-specific characteristics of the supported communications protocol.
- An interface component separate from the communications driver that interprets the message, executes the required code, and eventually returns data to the communications driver.
- A common file that all applications use to store the information on the queries to which they respond (i.e., REMOTE PROCEDURE file [#8994]).
- Architecture that supports multiple GUI and client front-ends.

This version of the Broker also includes the Broker Development Kit (BDK). The BDK provides **VISTA** application programmers with the following features:

- The capability to create GUI client/server **VISTA** applications using Inprise Delphi. The BDK provides the TRPCBroker component, which developers use in Delphi applications to execute remote procedure calls on **VISTA** M servers.
- Support for COTS/HOST client/server software using the Broker Dynamic Link Library (DLL).

The RPC Broker:

- Operates in a 32-bit environment (i.e., client workstations running Microsoft Windows 95 or Windows NT operating systems) while supporting **VISTA** applications previously developed in the 16-bit environment (e.g., PCMM).
- Provides support for single signon. Users need only sign on once when accessing both a **VISTA** roll-and-scroll (e.g., Lab, Pharmacy) and a **VISTA** client/server GUI-based application (e.g., PCMM, CPRS) on the same workstation, regardless of which application is started first.
- Provides new and enhanced Broker management and configuration tools (e.g., new debugging tools, new RPC BROKER SITE PARAMETERS file (#8994.1), enhanced Broker Listener).

How Does It All Work?

The process begins on a user's workstation (i.e., PC), running Microsoft Windows, which is either connected directly or remotely via a modem to a hospital's local area network (LAN). The workstation must be able to run some version of Transmission Control Protocol/Internet Protocol (TCP/IP). For more specific environment requirements, please refer to the *RPC Broker V. 1.1 Installation Guide*.



Currently only Winsock-compliant TCP/IP protocol is supported on the LAN or remotely as Point-to-Point Protocol (PPP) or Serial Line Internet Protocol (SLIP). You must use RAS (Remote Access Service) or Dialup Networking to connect to the server using PPP or SLIP. For the setup of RAS or Dialup Networking, please refer to the appropriate operating system's documentation.

When a user starts a **VISTA** program on the client, the program requests a connection with a server. The server is continuously running at least one Broker "Listener" job in the background whose sole purpose is to establish connections with clients.

Once the Listener receives a connection request, it does the following:

1. Validates the message.
2. Creates (spawns, jobs off) another process "Handler." The Handler process does the work to satisfy the client's requests.
3. Goes back to listening.

When the connection to the server is established, users who are not already logged into the server are asked to identify themselves by logging in with their Access and

Verify codes. With the implementation of single signon, users are considered already logged in to the server if they have previously logged in to a **VISTA** GUI or roll-and-scroll application that is still running on their workstation. After a successful login, the application is active on both the server and the client.

As you manipulate the interface, your client process is reading and writing data to the server. The reading and writing is carried out as messages traveling over the TCP/IP link. In the message sent to the server, client applications will include the name of the requested RPC to be activated and its associated parameters. These RPCs will be written in M and registered in a file containing available and authorized RPCs (i.e., REMOTE PROCEDURE file [#8994]). Upon receipt by the server, the message is decoded, the requested remote procedure call is activated, and the results are returned to the calling application.

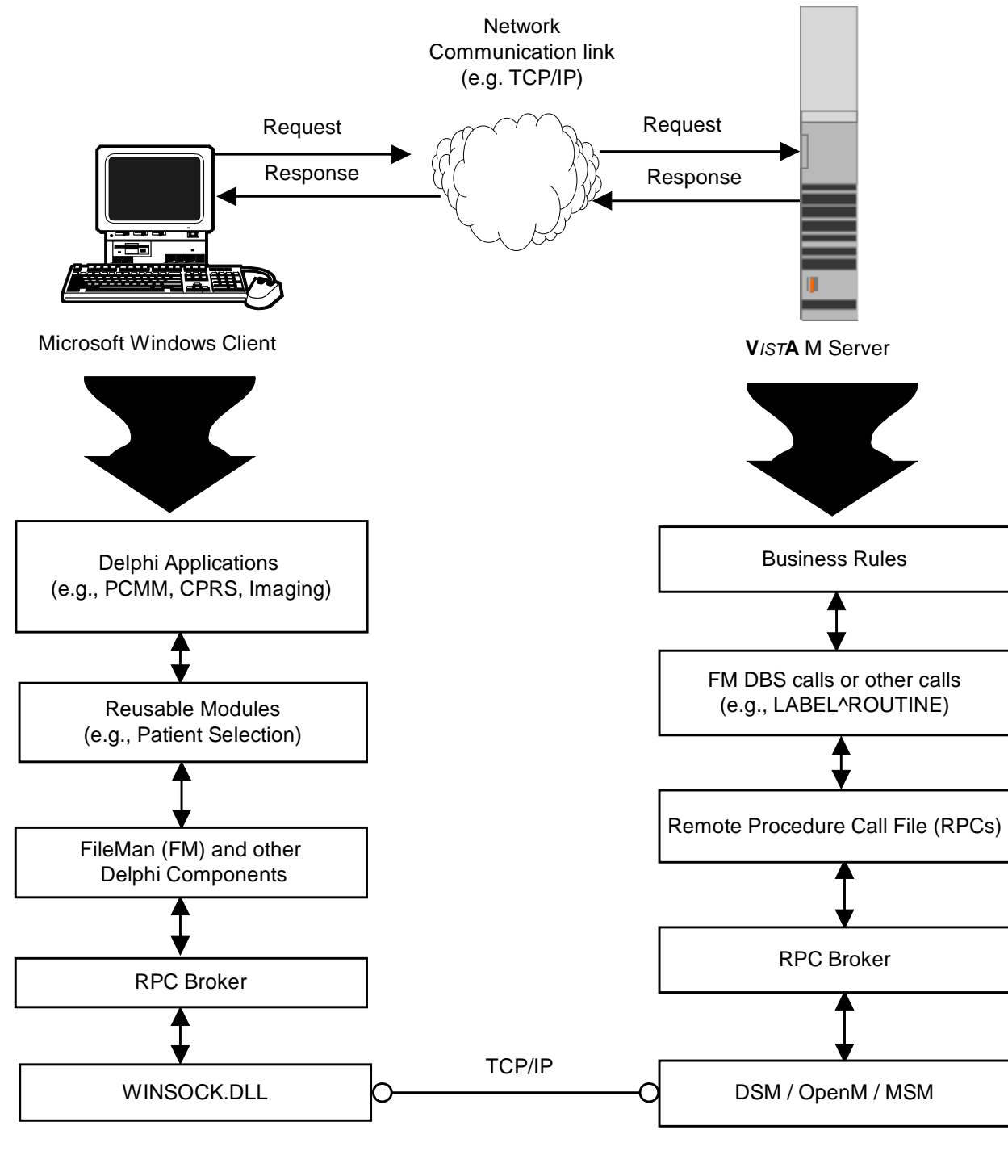
The server receives a message from the client and parses out the name of the remote procedure call and its parameters. The Broker module on the server looks up the remote procedure call in the REMOTE PROCEDURE file (#8994), verifies that the RPC is allowed to run in the context of the application, and executes the RPC using the passed parameters. At this point the server side of the application processes the request and returns the result of the operation. The result of the call contains either several values or a single value. If the operation is a query, then the result is a set of records that satisfy that query. If the operation is to simply file the data on the server or it is unnecessary to return any information, then, typically, notification of the success of the operation will be returned to the client.



This version of the RPC Broker supports messaging for non-Delphi client applications (e.g., Borland C++, Microsoft Visual Basic, or other COTS Windows-based products). The RPC Broker Version 1.1 supplies a set of functions providing a Dynamic Link Library (DLL) interface that allows non-Delphi applications to conform to the client side interface of the Broker. For more specific information about the Broker DLLs, please refer to the Online RPC Broker Developer's Guide (i.e., BROKER.HLP).

System Overview

The following diagram gives an overview of the **VISTA/Broker** environment:

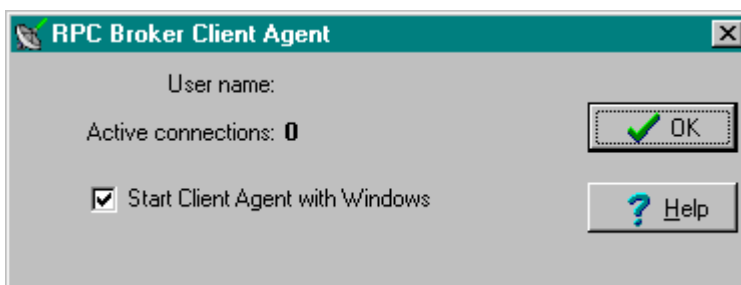


1. System Features

CLIENT FEATURES

RPC Broker Client Agent

The RPC Broker Client Agent program (i.e., CLAGENT.EXE) runs in support of the single signon process (a.k.a. auto signon). This program automatically and continuously runs in the background on the client workstation and normally should *not* be closed or shut down by the user. A satellite dish icon will be displayed in the System Tray indicating the Broker Client Agent is running. The icon will change when an active connection is made to the server, and a green line indicating an active connection will emanate from the satellite dish.



By double clicking on the Client Agent icon, you can see how many active connections are currently open. However, the "Active connections" count may include "orphan" connections that are no longer active. Use this count as an approximate count only.

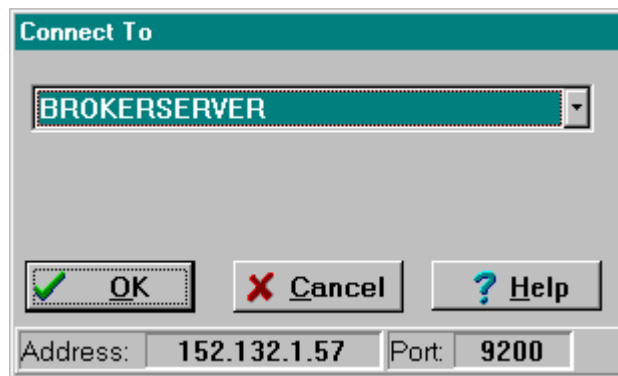


The "Start Client Agent with Windows" checkbox should be checked so that single signon, if allowed, will be operational. By default, this box is checked. However, if a particular workstation is not always connected to the network upon startup, you may wish to prevent Client Agent from starting automatically. You can always reset it to start automatically by starting Client Agent manually first and re-checking this check box.

"Connect To" Application Window

Upon logging in to a **VISTA** client/server application, users may be presented the "Connect To" application window displayed below. This application window may be used by Delphi **VISTA** client/server applications that wish to allow users to select a server name and associated port from a list of servers entered into the Windows Registry. For example, this can be useful when you want to run the application in either a Test or Production account.

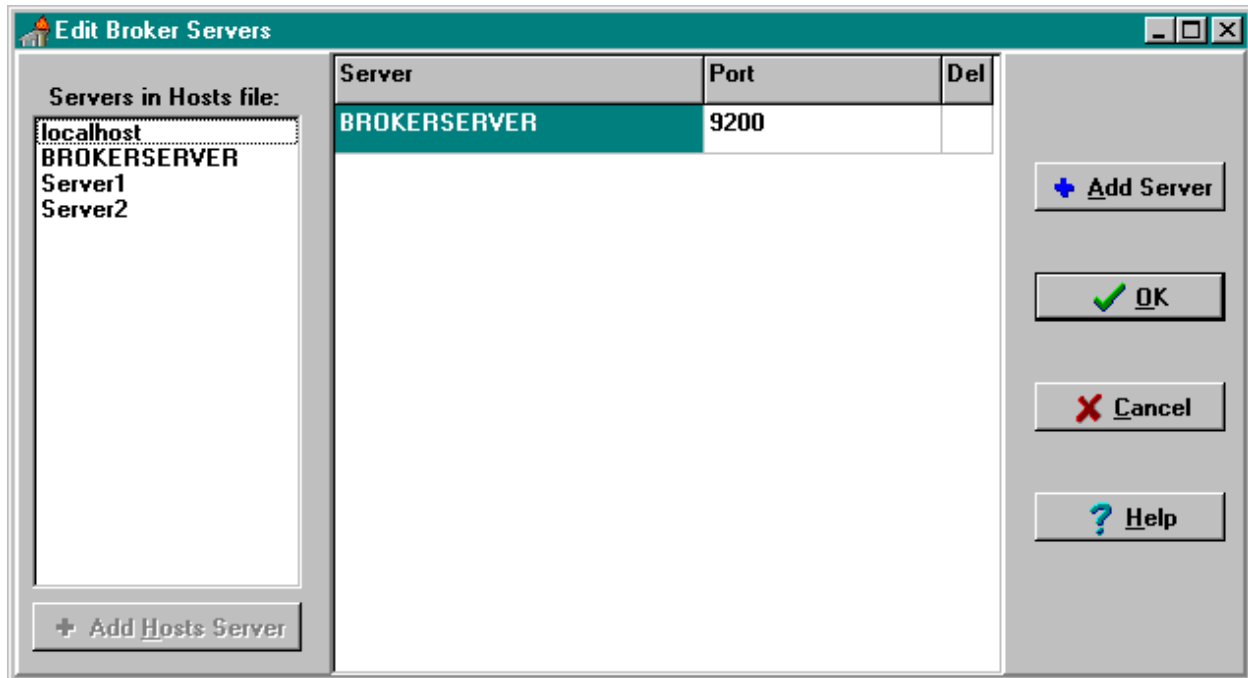
Server and Port Configuration Selection Application Window



To add additional server names and ports to the Windows Registry, please refer to the "Edit Broker Servers Program" topic below.

Edit Broker Servers Program

If someone in IRM wishes to add, modify, or delete servers and ports to be used by the Broker, they will run the Edit Broker Servers program (i.e., ServerList.EXE, see below) to modify or add Listeners/Ports to the Windows Registry. ServerList.EXE can be copied to any workstation for this purpose.



This program only displays HOSTS file entries; it does not edit the HOSTS file.

Adding Entries:

You are given two methods of adding new server entries to the Registry:

1. A list of servers available from your HOSTS file is displayed in the Servers in Hosts file list box on the left of the dialog box. Thus, you don't have to remember and type the server names yourself. Select one or more from the list and press the Add Hosts Server button. This creates the new grid line(s) with the server(s) selected automatically "stuffed" into the Server cell(s). Complete each new entry by typing in the appropriate port(s). When finished, press OK.
2. Alternatively, you can press the Add Server button on the right of the dialog box. This creates a new grid line. Enter the name of the server you want added. Complete each new entry by typing in the appropriate port(s). When finished, press OK. The program will attempt to resolve the server name to an IP address either through the Domain Name Service (DNS) or by looking it up in the HOSTS file on the client workstation. If this is successful, the new entry will be added to the Registry. If the server name cannot be resolved, an error message will be displayed and you will have to correct your entry.



Hint: If you're running a PC Network with Windows NT, the BROKERSEVER added to the Services list on the NT network will speed up

client access times (i.e., keeps it from having to do a double lookup with first IP then service, it merely looks at the Services list).

Modifying Entries:

In order to modify or change a server or port, simply place the cursor in the appropriate Server or Port field and make the change. When finished, press OK.



Server names must be resolvable through DNS or the HOSTS file.

Deleting Entries:

In order to delete a pre-existing entry just click in the Del column. An asterisk appears in the Del column signifying a deletion. Another click toggles the deletion off. When finished, press OK.

Stand-alone Programs and their Associated Help Files

Each of the following stand-alone Broker programs, distributed with this version of the Broker, have an associated help file that must reside in the *same* directory in order to provide online help for that particular stand-alone program:

Stand-alone Program	Associated Help File	Location
BROKERPROGPREF.EXE	BROKERPROGPREF.HLP	Programmer Workstation
CLAGENT.EXE	CLAGENT.HLP	End-user Workstation
RPCTEST.EXE	RPCTEST.HLP	End-user Workstation
SERVERLIST.EXE	SERVERLIST.HLP	Programmer Workstation

The installation of the Broker will automatically load these associated files into the same directory. If you choose to "export" a stand-alone program (e.g., SERVERLIST.EXE) to another client workstation, make sure you include its associated help file and place them both in the *same* directory.



*For more information on the **BROKERPROGPREF.EXE**, please refer to the **Online RPC Broker Developer's Guide** (i.e., **BROKER.HLP**).*

*For more information on the **CLAGENT.EXE**, please refer to the "RPC Broker Client Agent" topic above.*

*For more information on the **RPCTEST.EXE**, please refer to the Chapter 3, "Troubleshooting" in this manual.*

*For more information on the **SERVERLIST.EXE**, please refer to the "Edit Broker Servers Program" topic above.*

HOSTS File

The HOSTS file is an ASCII text file that contains a list of the servers and their IP addresses. However, use of the HOSTS file is *not* a requirement for the Broker. The use of the HOSTS file depends on the way the local area network (LAN) is implemented and managed at a site. Clients can bypass the HOSTS file and use DNS, DHCP (Dynamic Host Configuration Protocol), or WINS (Windows Name Service).

To modify or add servers to the HOSTS file, edit the file using a text editor (e.g., Microsoft Notepad).

The following table illustrates where you can find this file based on your client Windows operating system (OS):

Version of Windows OS	File (Location and Name)
Windows 95	C:\WINDOWS\HOSTS
Windows NT 3.51	C:\WINDOWS\SYSTEM32\DRIVERS\ETC\HOSTS
Windows NT 4.0	C:\WINNT\SYSTEM32\DRIVERS\ETC\HOSTS

A sample of the Windows 95 HOSTS file (i.e., C:\WINDOWS\HOSTS.SAM) is displayed below (modifications/additions made to this sample file are italicized):

Sample HOSTS File

```
# Copyright (c) 1994 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Chicago
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host
# name. The IP address and the host name should be separated by at
# least one space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#       102.54.94.97      rhino.acme.com      # source server
#       38.25.63.10      x.acme.com          # x client host
#
#      IP                Host
# ADDRESS              Name                Description
#      |                |                |
#      |                |                |
#      V                V                V
#
127.0.0.1      localhost      # loopback
192.1.1.1      BROKERSERVER    # Broker Server
```

The last entry in this file (i.e., BROKERSERVER) was added to the sample HOSTS file for illustration purposes. We recommend you put in an entry that points to the main server you intend using with the Broker the majority of the time (e.g., BROKERSERVER). **VISTA** applications can specify any server they wish.



A DHCPSEVER entry is still required for software that uses Version 1.0 of the Broker. You may want to create an additional entry for BROKERSERVER in your HOSTS file or DNS. However, *do not remove* the DHCPSEVER entry already present.

Adding Entries:

To add entries in the HOSTS file:

- a. Move the cursor to the end of the last line displayed in the file.
- b. Press the Enter Key to create a new line.
- c. On the new line, enter the desired IP address beginning in the first column, as described in the sample HOSTS file shown above. As recommended, add an appropriate IP address for the BROKERSERVER Host name as the next entry below "127.0.0.1".
- d. After typing the IP address, type at least one space and enter the Host name that corresponds to that IP address. As recommended, type in BROKERSERVER as the next entry below "loopback".

For example, the entry for a server at your site with an IP address of 192.1.1.1 would look like the following:

```
127.0.0.1    localhost    # loopback    <---existing entry
192.1.1.1    BROKERSERVER # Broker Server <---added entry
```

- e. Repeat steps a - d until you have entered all of the IP addresses and corresponding Host names you wish to enter.
- f. When your entries are complete, use Notepad's **File** | **Save** command to save the HOSTS file.



Do not save the HOSTS file with an extension!

- g. Close the HOSTS file.

Modifying Entries:

To modify entries in the HOSTS file:

- a. Move the cursor to the line to be modified.
- b. Modify the IP address, Host name, or both.
 - Make sure that at least one space separates the IP address from the corresponding Host name.
 - Make sure you have an entry for BROKERSERVER in this file.
- c. Repeat steps a - b until you have modified all of the IP addresses and corresponding Host names you wish to change.
- d. When your entries are complete, use Notepad's **File** | **Save** command to save the HOSTS file.



Do not save the HOSTS file with an extension!

- e. Close the HOSTS file.

What Happened to the Client Manager?

The Client Manager, previously distributed with version 1.0 of the Broker, is no longer used by this version of the Broker. In version 1.0 of the Broker, the Client Manager provided two types of services:

1. It was used to invoke the RPCBI.DLL.
2. It was used by developers to set programmer preferences for using the TRPCBroker component.

The RPCBI.DLL that was distributed with the RPC Broker V. 1.0 is no longer used, thus, the Client Manager is no longer required with this version of the Broker. Configuration of programmer preferences will now be done via the Broker Programmer Preferences Application Window.



For more information on the Broker Programmer Preferences Application Window, please refer to the "RPC Broker Programmer Preference Editor" topic in the Online RPC Broker Developer's Guide (i.e., BROKER.HLP).



The RPCBI.DLL and Client Manager (i.e., CLMAN.EXE) installed with Broker V. 1.0 must *not* be removed from the vista/broker directory on the client workstation. They are still required for 16-bit Broker-based applications created using version 1.0 of the Broker (e.g., PCMM).



What Happened to the VISTA.INI File?

The VISTA.INI file will no longer be used by applications built with Version 1.1 of the Broker. However, this file will continue to be used by applications built using version 1.0 of the Broker (e.g., PCMM). During the installation of the Broker, relevant data from the VISTA.INI file will be moved to the Windows Registry. Subsequent reads and writes will be done via the Registry.



The VISTA.INI file created with Broker V. 1.0 must not be removed from the Windows directory on the client workstation. It is still required for 16-bit Broker-based applications created using version 1.0 of the Broker (e.g., PCMM).

The following are a list of items from the present VISTA.INI file and their disposition with this version of the Broker:

VISTA.INI File Item	Disposition
ClientManagerPath ErrorRetry ClientManagerState	Client Manager items - <i>not</i> moved to the Registry.
IdeConnect ClearParameters ClearResults ListenerPort Server	Programmer items - moved to the Registry via a developer workstation installation (to be edited by the new Configuration form).
SignonPos SignonSiz IntroBackCol IntroTextFont	Signon items - moved to the Registry (these did <i>not</i> exist in version 1.0). These items will now be edited from the Signon form.  <i>For more information, please refer to the "Users Can Customize Signon Window" topic in Chapter 2 of this manual.</i>
HostsPath	No longer useful (i.e., Broker V. 1.1 Delphi code will <i>not</i> reference it).
[RPCBroker_Servers] section	Server/Port pairs - moved to the Registry via general workstation installations. These entries will now be edited via the Edit Broker Servers application.  <i>For more information, please refer to the "Edit Broker Servers Program" topic in this chapter.</i>

SERVER FEATURES

Menu for System Managers

Patch XWB*1.1*9 introduces a new menu (XWB MENU) for system managers:

Select RPC Broker Management Menu Option:

```
RPC Listener Edit
Start All RPC Broker Listeners
Stop All RPC Broker listeners
```

Broker Listeners and Ports

You can run:

- A single Broker Listener, running on any available port.
- *Multiple* Broker Listeners running on the same IP address/CPU, but listening on *different* ports.
- *Multiple* Broker Listeners in the *same* UCI-volume, but on *different* IP addresses/CPUs, listening on the *same* port (or on different ports).

So, for example, to run one listener in a Production account and another in a Test account, on the same IP address/CPU, you must configure them to listen on different ports (e.g., 9200 for production and 9201 for Test). If, on the other hand, you are running the listeners on different IP addresses/CPUs, the ports can be the same (e.g., one Broker Listener on every system, listening on port 9200).

You need to configure your clients to connect to the appropriate listener port on your M server. While 9200 has been used as a convention for a Broker application service port, you can choose any available port greater than 1024 (sockets 1 to 1024 are reserved for standard, well-known services like SMTP, FTP, Telnet, etc.)

OBTAINING AN AVAILABLE LISTENER PORT (FOR ALPHA/VMS SYSTEMS ONLY)

Port selections conflict only if another process on the same system is using the same port. To list the ports currently in use on OpenVMS systems, use the DCL command:

```
$ UCX SHOW DEVICE_SOCKET
```

Device_socket	Type	Local	Port Remote	Service	Remote Host
bg3	STREAM	5001	0	HL7	0.0.0.0
bg23	STREAM	9700	0	Z3ZTEST	0.0.0.0
bg24	STREAM	9600	0	ZSDPROTO	0.0.0.0

For example, if 9200 shows up in the Local Port column, some other application is already using this port number and you should choose another port.

Starting And Stopping Listeners

TO START ALL LISTENERS

To start all listeners (i.e., those configured in the RPC Broker Site Parameters file to automatically start), use the "Start All RPC Broker Listeners" option (introduced by patch XWB*1.1*9). This option first **stops** any of these listeners that may be running, and then starts all of them up. TaskMan must be running.



DSM sites must run TaskMan in a DCL context, to start Listeners on any other node than the current ones. For more information on running TaskMan in a DCL context on DSM for OpenVMS, please see the Kernel Systems Manual

TO CONFIGURE LISTENERS FOR AUTOMATIC STARTUP

To configure a given listener for startup by the "Start All RPC Broker Listeners" option, enter YES in the "Controlled By Listener Starter" field in the RPC Broker Site Parameters file for that listener. For more information, please see the "RPC Broker Site Parameters File" section of this chapter.

TO STOP ALL RUNNING LISTENERS

To stop all running listeners (but, only those configured in the RPC Broker's site parameters to automatically start), use the "Stop All RPC Broker Listeners" option.



It is important to stop all Listeners before shutting down the system!

TO START UP A SINGLE LISTENER DIRECTLY

- a. Enter the following at your M server's M prompt:

```
>D STRT^XWBTCP(Listener port)
```

TO STOP A SINGLE LISTENER DIRECTLY

- a. Enter the following at your M server's M prompt:

```
>D STOP^XWBTCP(Listener port)
```

Note: If you want to restart this listener after stopping it, and other listeners are running on your system, start the listener up directly (see above) rather than via the "Start All RPC Broker Listeners" option (which first stops all listeners before restarting them).

TO TASK THE XWB LISTENER STARTER OPTION FOR SYSTEM STARTUP

The XWB LISTENER STARTER option (which starts all configured Broker Listeners at one time). can be tasked to automatically start all of the Listener processes you need when TaskMan starts up, such as after the system is rebooted or configuration is restarted.



DSM sites must have TaskMan started via DCL, in order to use the XWB LISTENER STARTER option to automatically start Listener processes.

To automatically start the Listener(s) when TaskMan is restarted (i.e., in addition to the entries in the RPC BROKER SITE PARAMETERS file [#8994.1]), enter the XWB LISTENER STARTER option in the OPTION SCHEDULING file (#19.2). Schedule this option with SPECIAL QUEUING set to STARTUP. You can do this by using the TaskMan option: Schedule/Unschedule Options:

```
Select Systems Manager Menu Option: TASKMAN Management

Select Taskman Management Option: SCHedule/Unschedule Options

Select OPTION to schedule or reschedule: XWB LISTENER STARTER <RET>
Start All RPC Broker Listeners
...OK? Yes// <RET> (Yes)
(R)

                                Edit Option Schedule
Option Name:   XWB LISTENER STARTER
Menu Text:    Start All RPC Broker Listeners      TASK ID:

-----

QUEUED TO RUN AT WHAT TIME:

DEVICE FOR QUEUED JOB OUTPUT:

QUEUED TO RUN ON VOLUME SET:

RESCHEDULING FREQUENCY:

TASK PARAMETERS:

SPECIAL QUEUEING:   STARTUP
```



If you are an MSM 4.3.0 site or greater and using MSERVER instead of the Broker Listener, the XWB LISTENER STARTER option is not applicable to your site. Please refer to the "MSM for NT 4.3.0 MSERVER Replaces RPC Broker Listener Process" topic below.

RPC BROKER SITE PARAMETERS File

The RPC BROKER SITE PARAMETERS file (#8994.1) contains one top-level entry, whose .01 field is a pointer to the Domain file. When the RPC Broker is installed, you create this top-level entry and assign the proper Domain name.

The site parameters in this top-level entry pertain to listeners. For each listener that you plan to run on your system, you should make an entry for that listener in the site parameters.

EDITING THE LISTENER SITE PARAMETERS

For each listener you plan to run on your system, there should be an entry in the RPC Broker Site Parameters file. To create or edit listener entries, use the "RPC Listener Edit" option.

The "RPC Listener Edit" option first prompts you to select a Box-Volume Pair entry. Then, within each Box-Volume Pair entry (representing the volume set and system the listener should run on), you can configure one or more listeners:

```
Select RPC BROKER SITE PARAMETERS DOMAIN NAME: YOURSITE.VA.GOV
...OK? Yes//    (Yes)

Select BOX-VOLUME PAIR: KDE:ISC6A2//
BOX-VOLUME PAIR: KDE:ISC6A2//
Select PORT: 9500//
PORT: 9500//
STATUS: STARTING//
CONTROLLED BY LISTENER STARTER: YES//
```

The meaning of the site parameter fields for a given listener entry is as follows:

Field	Meaning
Box-Volume Pair	Choose the Box-Volume pair representing one of the systems supporting "this" account, and on which a listener should run.
Port	The port the listener will listen on.
Status	Ordinarily should not be edited (Use the "Start All RPC Broker Listeners" and "Stop All RPC Broker Listeners" options to start and stop listeners.)
Controlled By Listener Startup	If the listener should be started by the "Start All RPC Broker Listeners" option (XWB LISTENER STARTER), set this field to YES. Otherwise, set to NO.

Integrated Single Signon for Multiple User Sessions

This version of the RPC Broker supports Kernel's single signon from a client workstation to the server. Users need only sign on once (i.e., enter their Access and Verify codes) when accessing both a **VISTA** roll-and-scroll (e.g., Lab, Pharmacy) and a **VISTA** client/server GUI-based application (e.g., PCMM, CPRS) on the same workstation, regardless of which application is started first. Once logged into the server, the user will *not* be asked to re-enter their Access and Verify codes for any subsequent **VISTA** applications they may start.



Single signon is facilitated on the client side by the Broker Client Agent application (CLAGENT.EXE) and is only available for Telnet-based sessions in the roll-and-scroll environment.

ENABLING/DISABLING SINGLE SIGNON

Control of the single signon functionality is maintained and administered on the server for both **VISTA** client/server applications (i.e., GUI) and the roll-and-scroll environment (i.e., terminal sessions). In support of that functionality a new field, DEFAULT AUTO SIGN-ON, was added to the KERNEL SYSTEM PARAMETERS file (#8989.3) and AUTO SIGN-ON was added to the NEW PERSON file (#200). The valid values for these fields are Yes, No, or Disabled.

These new fields, in conjunction with the existing multiple signon fields, give the sites control of the implementation of single signon for users in both the GUI and roll-and-scroll environments. The values in the AUTO SIGN-ON and MULTIPLE SIGN-ON fields in the NEW PERSON file (#200) take precedence over the values in the DEFAULT AUTO SIGN-ON and DEFAULT MULTIPLE SIGN-ON fields in the KERNEL SYSTEM PARAMETERS file (#8989.3). Therefore, the fields in the NEW PERSON file are checked first. If the user fields in the NEW PERSON file are null, the values in the KERNEL SYSTEM PARAMETERS file will be used.



The AUTO SIGN-ON field in the NEW PERSON file and the DEFAULT AUTO SIGN-ON field in the KERNEL SYSTEM PARAMETERS file are initially set to null.



Single signon is *not* supported on MSM systems. All MSM sites must set the DEFAULT AUTO SIGN-ON field in the KERNEL SYSTEM PARAMETERS file (#8989.3) to Disabled.



If a user is *not* allowed multiple signons, they will only be allowed to initiate a *single* session (i.e., automatically disallowing single signon).

Example 1:

If a user has an active **VISTA** session and has the following characteristics:

- Allowed multiple signons (i.e., the MULTIPLE SIGN-ON field in the NEW PERSON file (#200) is set to Yes)
- Allowed auto signon (i.e., the AUTO SIGN-ON in the NEW PERSON file (#200) is set to Yes)

They will be allowed to start another **VISTA** session *without* having to re-enter their Access/Verify codes.

Example 2:

If a user has an active **VISTA** session and has the following characteristics:

- Allowed multiple signons (i.e., the MULTIPLE SIGN-ON field in the NEW PERSON file (#200) is set to Yes)
- Not allowed auto signon (i.e., the AUTO SIGN-ON field in the NEW PERSON file (#200) is set to No)

They will be allowed to start another **VISTA** session, however, they *must* re-enter their Access and Verify codes.

The following table can be used as a guide to control multiple signons and single signon (a.k.a. auto signon) for some typical situations:

Description	*User Settings	**System Settings
Multiple Signon:		
Disallow <i>all</i> users from having multiple signons	No/Null	No
Allow <i>individual</i> users to have multiple signons	Yes	No
Allow <i>all</i> users to have multiple signons	Yes/Null	Yes
Auto Signon:		
NOTE: With the exception for disabling auto signon, the following settings are only affective when users are allowed multiple signons.		
Stop auto signon	Any Value	[†] Disabled
Allow <i>individual</i> users to have auto signon	Yes	No
Disallow <i>individual</i> users from having auto signon	No	Yes
Allow <i>all</i> users to have auto signon	Yes/Null	Yes

[†]Sites may choose to disable single signon (stops calls to the Broker Client Agent) for all users in the following situations:

- Network problems
- Broker not installed
- During installation of the Broker



Single signon is *not* supported in MSM systems. All MSM sites must set the DEFAULT AUTO SIGN-ON field in the KERNEL SYSTEM PARAMETERS file (#8989.3) to Disabled.

*User Settings refers to the NEW PERSON file (#200) and the following fields:

- MULTIPLE SIGN-ON (#200,200.04)
- AUTO SIGN-ON (#200,200.18)



The User Settings override the System Settings *except* when *disabling* single signon!

**System Settings refers to the KERNEL SYSTEM PARAMETERS file (#8989.3) and the following fields:

- DEFAULT MULTIPLE SIGN-ON (#8989.3,204)
- DEFAULT AUTO SIGN-ON (#8989.3,218)

RPC Broker Message Structure

The messages that are sent from a server to a client contain either several values or a single value. Presently, the RPC Broker messages are bound by the Windows WINSOCK.DLL specifications and the size of the symbol table. The server receives a message from the client and parses out the name of the remote procedure call and its parameters. The Broker module on the server looks up the remote procedure call in the REMOTE PROCEDURE file (#8994) and executes the RPC using the passed parameters. At this point the server side of the application processes the request and returns the result of the operation. If the operation is a query, then the result is a set of records that satisfy that query. If the operation is to simply file the data on the server or it is unnecessary to return any information, then, typically, notification of the success of the operation will be returned to the client.

The basic RPC Broker message structure consists of the following:

- A header portion (which includes the name of the remote procedure call)
- The body of the message (which includes descriptors, length computations, and M parameter data)

Client/Server Timeouts

The issue of timeouts is complex in a client/server environment. Because the user may be working with applications that rely solely on the client, long periods of time may elapse that the server would traditionally have counted against the user's time-out.

To address this issue, patch XWB*1.1*6 was issued. It institutes a "keep-alive" timer that is compiled into client applications. Through monitoring this keep-alive timer, it is able to eliminate "ghost" server Broker jobs for which there is no longer a client application, based on the keep-alive timer rather than on user activity.

"Ghost" server jobs occur when client processes are ended in a non-standard way - for example, by pressing the PC's reset button. Prior to this patch, these jobs would wait for 10 hours to receive data from the client application that no longer exists.

In order to let the server know that the client application is still active, applications compiled with the client portion of patch XWB*1.1*6 initiate a periodic, background contact with the server. This 'polling' of the server by the client resets the timeout so that the server job is not stopped when the client still exists. Any client application compiled with the RPCBroker component distributed with this patch automatically polls. No developer or user intervention is necessary, and this polling activity affects neither the application nor the user.

The length of the timeout is controlled by a new field in the Kernel System Parameters file: BROKER ACTIVITY TIMEOUT. That field is distributed by Kernel patch XU*8.0*115 with a default value of approximately 3 minutes. By setting the timeout to a duration much shorter than 10 hours, the ghost jobs are eliminated quickly if the client application is no longer running. See the help for the BROKER ACTIVITY TIMEOUT field for advice regarding changing the value for this field.

The server portion of this patch is backwards compatible with client applications compiled with previous versions of the Broker. So, client applications do not have to be recompiled when this patch is installed on the server. The server retains a 10 hour timeout for those client applications compiled with previous Broker versions; that is, they continue to work as they did before the patch is installed.

Note: The server side of this patch is effective only for client applications (like CPRS-GUI) that have been recompiled with the Broker Development Kit portion of patch XWB*1.1*6. So, installing the server patch alone does not eliminate the ghost jobs for client applications that have not been upgraded.

Load Balancing on Alpha Systems

The Broker, like any telnet or IP process, can be load balanced on DSM Alpha systems if UCX 4.1 is running. The actual steps on configuring UCX for load balancing can be acquired from the ALPHA/AXP technical support group and will not be discussed here.

Multiple Broker servers can run on the same port as long as the machine IP addresses are unique. This is *not* a Broker requirement; it is a TCP/IP requirement. This capability is necessary for UCX load balancing. The multiple servers will receive a common alias that will be the connection destination.

In UCX, you should use the BIND alias:

For example:

```
UCX> show host vista.sitename.med.va.gov

      BIND database

Server:  152.999.999.xxx      999TNG

Host address      Host name
152.999.999.yy1   VISTA.SITENAME.MED.VA.GOV
152.999.999.yy2   VISTA.SITENAME.MED.VA.GOV
152.999.999.yy3   VISTA.SITENAME.MED.VA.GOV
152.999.999.yy4   VISTA.SITENAME.MED.VA.GOV
152.999.999.yy5   VISTA.SITENAME.MED.VA.GOV
```

In order to use load balancing, your client workstation needs to have DNS enabled and pointing to the IP address of the DNS server in the list. The Broker on the client will use the PC's DNS or HOSTS file to resolve the BROKERSERVER host name. In the previous example, the first DNS server is 152.999.999.xxx.

For example, If you want PCMM GUI to be "balanced," use the Edit Broker Servers program to edit the servers in the Registry and add in the alias VISTA.sitename.MED.VA.GOV.



For more information on adding servers to the Registry, please refer to the "Edit Broker Servers Program" topic in this chapter.

You don't want the alias in the HOSTS file because the HOSTS file is for static bindings only. If you want to put the alias in the HOSTS file, then you will have to make sure that the DNS server is first in the DNS list. Thus, when the user selects

the BIND alias, the DNS will resolve it to one of the unique IP addresses and *not* to the HOSTS static assignment.



DSM sites do not need to have TaskMan started in a DCL context in order to use load balancing.

MSM for NT 4.3.0 MSERVER Replaces RPC Broker Listener Process

MSM for NT in version 4.3.0 introduced a new generic TCP/IP Listener MSERVER. MSERVER is an actual routine in the Manager's UCI that runs and listens to all of the ports that you specify in SYSGEN. When a connection is established to one of these ports, MSERVER launches your code at some TAG^ROUTINE that you specify. This is similar to VMS' UCX utility. If you are running a Beta version of MSM 4.3 for NT, we encourage you to upgrade to 4.3.0. Once you upgrade, you should stop using the RPC Broker Listener and switch to the MSERVER.

Before MSERVER can be used, it must be configured using SYSGEN. The following is an example from the CIO Field Office in San Francisco:

1. In the Manager's UCI run SYSGEN.
2. In sequence pick the following options:
 - a. 3 - Edit Configuration Parameters
 - b. 15 - Network Configuration
 - c. 12 - User-Defined Services
3. Enter an unused index number and set it up as indicated below:
 - a. Enter Service Name <RPC BROKER>: **RPC BROKER**
 - b. Enter Routine Reference <MSM^XWBTCPC>: **MSM^XWBTCPC**
 - c. Select UCI <VAH,MNT>: **VAH,MNT**
 - d. Enter Partition Size <80>: **80**
 - e. Enter Password <>: **<RET>**
 - f. Enter TCPIP Port Number <9200>: **9200**
 - g. Autostart? <YES>: **YES**

However, specifying YES for Autostart is not enough for MSERVER to start up when MSM is started. You must also add MSERVER to the Automatic Startup List as indicated in Steps 4-6 below.

4. To schedule MSERVER to start automatically when MSM comes up, go to the SYSGEN main menu.
5. In sequence pick the following options:
 - a. 3 - Edit Configuration Parameters
 - b. 3 - Autostart and Automounts
 - c. 5 - Automatic Partition Startup
6. Enter an unused index number and set it up as indicated below:
 - a. Enter UCI name <MGR>: **MGR**
 - b. Enter Entry Reference <STARTUP^MSERVER("RPC BROKER")>:
STARTUP^MSERVER("RPC BROKER")
 - c. Enter partition size <SYSTEM>: **SYSTEM**

To start MSERVER manually, in the Manager's UCI, type the following at the programmer prompt:

```
>J STARTUP^MSERVER("RPC Broker")
```

To shut down MSERVER in the Manager's UCI, type the following at the programmer prompt:

```
>J SHUTDOWN^MSERVER("RPC Broker")
```

If you're successfully using MSERVER, discontinue using the Broker Listener. That means you shouldn't use the STRT^XWBTCP and STOP^XWBTCP entry points or the RPC BROKER SITE PARAMETERS file previously discussed.

|

2. Security

SECURITY FEATURES

Security in distributed computing environments, such as in client/server systems, is much more complicated than in traditional configurations. Although it is probably impossible to protect any computer system against the most determined and sophisticated intruder, the RPC Broker implements robust security that is transparent to the end user and without additional impact on IRM.

Security with the RPC Broker is a four-part process:

1. Client workstations must have a valid connection request
2. Users must have valid Access and Verify codes
3. Users must be valid users of a **VISTA** client/server application
4. Any remote procedure call must be registered and valid for the application being executed

VALIDATION OF CONNECTION REQUEST

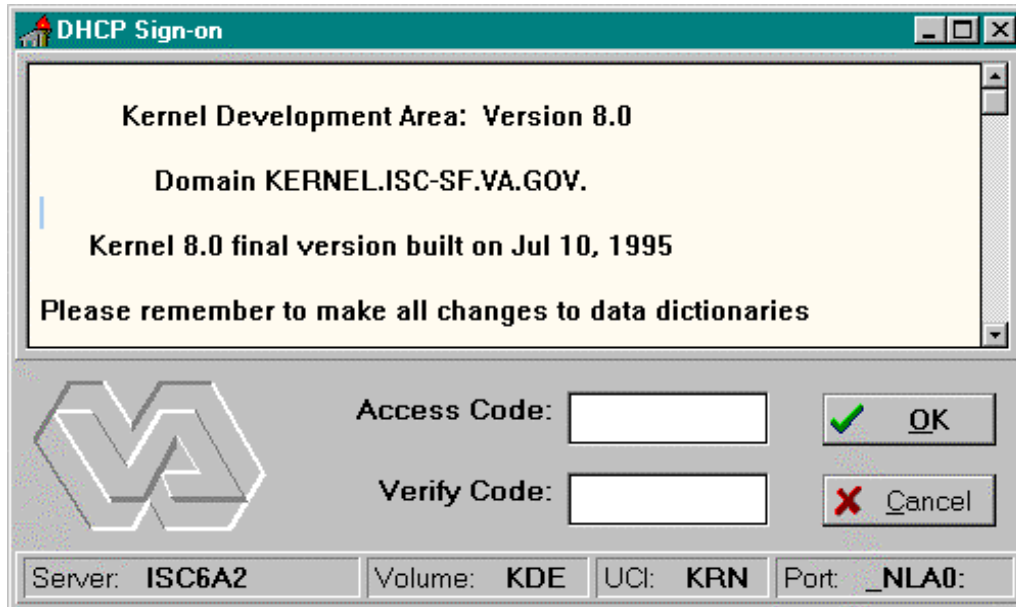
An enhancement to security has been included with this version of the Broker. Before the Broker Listener jobs off a Handler for a client, it checks the format of the incoming connection request. If the incoming message does not conform to the Broker standard, the connection is closed. This serves as an early detection of impostors and intruders.

VALIDATION OF USERS

The GUI signon application is integrated with the RPC Broker interface. This GUI signon is invoked when the client application connects to the server. The signon application automatically prompts users for their Access and Verify codes if they are not already signed on to a **VISTA** application.

A sample of the GUI signon window integrated with the RPC Broker is illustrated below:

Sample Signon Security Application



i *This version of the Broker supports single signon. For more information regarding single signon, please refer to the description in Kernel Patch #59 (i.e., XU*8*59).*

Client/server applications are a "B" (i.e., Broker) type of option in the OPTION file (#19). Users must have the client/server application option assigned to them like any other assigned option in **VISTA**. It can be put on their primary menu tree or as a secondary option/menu as part of their suite of permitted options. The client/server application will only run for those users who are allowed to activate it.

i *The client/server application options will not be displayed in a user's menu tree.*

Kernel's Menu Manager verifies that users are allowed access to a **VISTA** application or option with the following process:

1. Users start a **VISTA** program.
2. The RPC Broker in the client application invokes the GUI signon when connecting to the server.
3. Users sign on to the server via the Kernel signon process (see).
4. If authorized, the user is granted access to the server, otherwise an error message is returned. This serves as an initial security check.

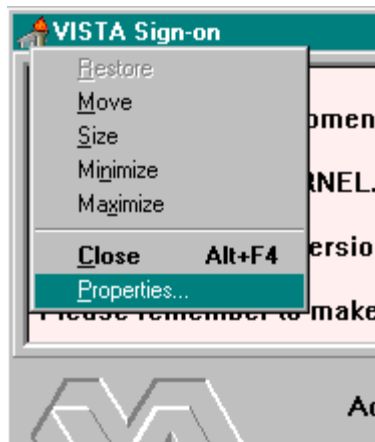


For more information on Access and Verify codes or the Kernel signon process in general, please refer to the Kernel Systems Manual.

Users Can Customize Signon Window

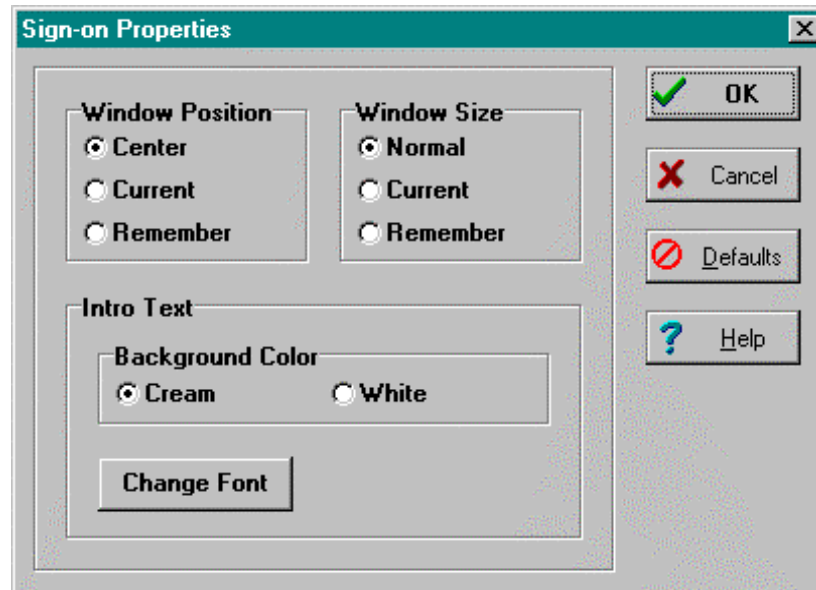
When a **VISTA** program on the client connects to the server, the **VISTA** signon window is displayed for the user to identify and authenticate himself on the server. This window's System menu (i.e., see the upper left corner of the application window displayed in) has a "Properties..." item:

Sign-on Properties Menu



When this item is selected, the user is presented with the following configuration dialogue box:

Sign-on Properties Application Window



Using the form displayed in , users can control the appearance of the signon window by modifying the following characteristics:

- Position of the signon window
- Size of the signon window
- The appearance of the introductory text

Window Position

The window position can be one of the following:

- | | |
|-------------------------|--|
| Center (default) | The window will always appear in the center of the screen. |
| Current | The current position of the window will be saved and used in the future. |
| Remember | Each time the window is used and closed, it will record its position and open in that same place the next time it is used. |

Window Size

The window size can be one of the following:

Normal (default)	The size of the window as it was designed. Typically, this is 500 pixels wide by 300 pixels high.
Current	The current size of the window will be saved and used in the future.
Remember	Each time the window is used and closed, it will record its size and open with the same size the next time it is used.

Introductory Text

The introductory text has a couple of settings users can control:

Background Color:

Cream (default)	According to the VA GUI conventions, this is the background color that should be used with text that users <i>cannot</i> edit.
White	For clarity and brightness.

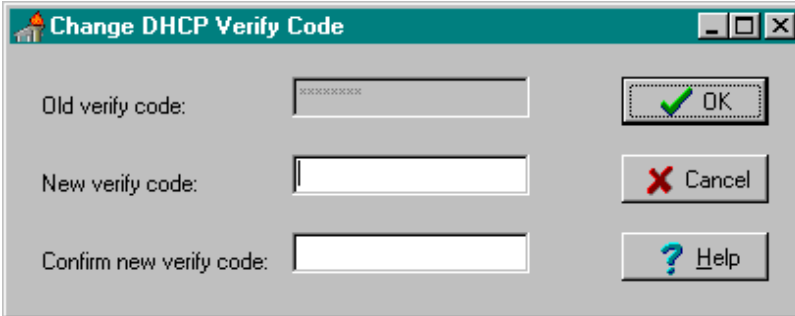
Font:

When users press the "Change Font" button they are presented with a Font form that can be used to change the font face, style, size, and color of the introductory text of the signon window.

Change VISTA Verify Code Component

This version of the Broker includes a new Change VISTA Verify Code dialog for the client workstation. After a user signs onto the server, if their Verify code has expired, the user is automatically prompted with the following message: "You must change your Verify code at this time." Once the user presses the OK button they are presented with the Change VISTA Verify Code window as displayed below:

Change VISTA Verify Code Dialog

The image shows a Windows-style dialog box titled "Change DHCP Verify Code". It has a teal header bar with a small icon on the left and standard window controls (minimize, maximize, close) on the right. The main area is light gray and contains three input fields and three buttons. The first row is labeled "Old verify code:" and has a text box filled with asterisks. The second row is labeled "New verify code:" and has an empty text box. The third row is labeled "Confirm new verify code:" and has an empty text box. To the right of the input fields are three buttons: "OK" with a green checkmark icon, "Cancel" with a red X icon, and "Help" with a blue question mark icon.

The old Verify code will appear as asterisks () in a grayed-out box.*

Users must then do the following:

- Enter their new Verify code
- Confirm their new Verify code

In the future, users will be able to invoke this dialog whenever they choose to modify their Verify code. For now, users who wish to change their Verify code prior to expiration may do so via the Edit User Characteristics option in the roll-and-scroll environment.

VALIDATION OF RPCs

The RPC Broker security allows any RPC to run when it is properly registered to the **VISTA** client/server application. The Broker on the server along with the Kernel's Menu Manager determines which application a user is currently running. Menu Manager determines if a user is allowed to run this application or option by the following process:

1. A remote procedure call is sent by a client application and is received by the RPC Broker on the server.
2. The Broker verifies that the RPC is "registered" to the application that the user is currently running, *prior* to executing the remote procedure call (RPC).

The application being run is designated by a "B"-type option in the OPTION file (#19). The application must specify the option and that option *must* be in a user's menu tree.



For more information on registering an RPC to a package, see the topic entitled "RPC Security: How to Register An RPC" in the RPC Broker V. 1.1 Getting Started with the Broker Development Kit manual.

3. Menu Manager checks if the RPC is registered for this package option. If not properly registered, Menu Manager will return a message explaining why the RPC is not allowed.
4. The Broker on the server executes the RPC if it is registered, otherwise it is rejected.

SAMPLE SECURITY PROCEDURES

The security steps each client user will follow and the intermediate client/server security processes are described in the following example:

Step Description

1. The user starts a **VISTA** program on the client. For this example, the user clicks on the Patient Care Management Module (PCMM) application icon.
2. The user must sign on to the server through the GUI signon window on the client using their Access and Verify codes (see) invoking the Kernel signon process.
3. The Menu Manager on the server verifies the user is allowed access to the "B"-type option requested by PCMM.
4. The Menu Manager on the server verifies the option is a "client/server" type option and the requested RPC is in that option's RPC multiple.
5. If all of the previous steps complete successfully, the application RPC is launched.

SECURITY FEATURES TASKS SUMMARY

The following table summarizes required security tasks:

Security Task	Completed By
Verify valid connection request	RPC Broker
Verify valid user	Kernel Signon
Verify user is authorized to run this package	RPC Broker & Menu Manager
Verify an RPC is registered to an application	RPC Broker & Menu Manager
Application - RPC Registration	KIDS



To reiterate, an RPC is only allowed to run within the context of an application with which it is registered. Users are only able to run the server side of the application that was installed on the server by IRM.



For each release of the RPC Broker, the RPC Broker Development Team will continuously strive to implement the most complete, robust, and flexible security available at the time.

3. Troubleshooting

TEST THE BROKER USING THE RPC BROKER DIAGNOSTIC PROGRAM

This version of the Broker includes a new diagnostic tool for the client workstation. This tool can be used to verify and test the Broker client/server connection and signon process. This program (i.e., RPCTEST.EXE) also displays specific information about the client workstation that can be useful to IRM personnel when trying to determine and/or correct any problems with or to test the Broker.

It displays the following information:

- Default workstation information that includes the Name and IP Address
- Local connection information that includes the Name, Client IP, and Current Socket
- **VISTA** user information that includes the Name and Last SignOn Date/Time
- Remote connection information that includes the Server, Port, IP Address, and Operating System Version information
- A color coded Link State indicator that shows the status of your connection (i.e., red = no link/connection, yellow = attempting link/connection, and green = successful link/connection)

When you run the diagnostic program (i.e., RPCTEST.EXE), the following application window will be displayed:

RPC Broker Connection Diagnostic Program

Vista RPC Broker Information Display [32-Bit]

Default WorkStation Info
 Name: isf-clarked.med.va.gov
 IP Address: 152.132.1.179

DHCP User Information
 Name:
 Last SignOn Date/Time:

Local Connection Info
 Name:
 Client IP:
 Current Socket:
 Broker State:

Remote Connection Info
 Server: BROKERSEVER
 Port: 9200
 IP Addr:
 OS Ver:
 Job ID:

Link State
☒

Buttons: Test Link, Log On, Log Out

You should verify that the connection from the client workstation to the server is functioning correctly. For example:

- Try logging on to the server by pressing the "Log On" button, you will be presented with the signon window. The Link State indicator will change from red to yellow to green as you progress through the connection process.
- Test various connections by changing the server and port information under the "Remote Connection Info" block. To verify the connection process is working properly, try logging on to known servers and ports with Listeners running.

You can also use this tool to resolve a server address without having to log on to the server. Type in a server name in the "Server" box located in the Remote Connection section of the dialog box and press the enter key. If the server can be found, the IP address will be displayed in the "IP Addr" box in that same section.

If you encounter an error while testing the Broker, make sure you check the following:

- Is the Broker Listener running on the specified port? If not, start the Broker Listener on the specified port.



For more information on starting the Broker Listener, please refer to the "Broker Listeners and Ports" topic in Chapter 1 of this manual.

- Have you installed all current Kernel, Kernel Toolkit, and VA FileMan patches? If not, you must install all required patches (see the *RPC Broker V. 1.1 Installation Guide*).
- Did you change the IP address for BROKERSERVER in the HOSTS file in this session? If the IP address and server name are not resolvable, you need to correct the entry.



Your site can use the HOSTS file or DNS to resolve IP addresses and server names. If the HOSTS file is not supported in your LAN, then you will need to work with the DNS database and see if the value returned by the DNS query really identifies the machine where the listener is running.

- Is the IP address resolvable for the BROKERSERVER listed under the TCP/IP Server? If not, edit the HOSTS file in your Windows directory and correct the IP address for the BROKERSERVER or resolve the IP address with DNS.
- Does the TCP/IP address (used in the HOSTS file) correspond to the IP address that is owned by the node used to start up the Broker Listener? If you have several nodes that can service your Test/Production account, you must make sure that the one used to start up the Listener is the one being referenced in the HOSTS file.

VERIFY AND TEST THE NETWORK CONNECTION

To detect and avoid network problems, do the following:

1. First, make sure you actually have TCP/IP running correctly on your workstation.

At the DOS prompt type PING nnn.nnn.nnn.nnn to the server host to which you are trying to connect (where nnn.nnn.nnn.nnn equals the IP address of the server). For example:

```
C:\>PING 127.0.0.1
```

Alternatively, you can PING the same server name you are trying to connect to or resolve (e.g., BROKERSERVER). For example:

```
C:\>PING BROKERSERVER
```



"PINGing" is a way to test connectivity. PINGing sends an Internet Control Message Protocol (ICMP) packet to the server in question and requests a response. It verifies that the server is running and the network is properly configured.

- If the host is unreachable, there is a network problem and you should consult with your network administrator.
 - If you get a time-out, it may be your network configuration on the client workstation, proceed to Step 2.
 - If the server is reachable, proceed to Step 4.
2. Check the properties of the WINSOCK.DLL on the client workstation and make sure it's the correct version. Install the latest Service Pack.
 3. Make sure that the files on the client are in the correct directories. In Windows 95, the WINSOCK.DLL expects the HOSTS file to be located in the WINDOWS root directory. You should only have one copy each of the WINSOCK.DLL and the HOSTS file on the client. (However, there may be a second copy that WIN95 keeps in the WINDOWS/SYSBCKUP directory). If Windows 95 detects that some of its core files have been overwritten with older versions, supposedly it will automatically update files on reboot.
 4. Make sure that all of the client workstation TCP/IP settings are correct in the network properties. Typo's, etc. can be a real problem, as can gateways, DNS servers, etc. Try removing items in your WINS configuration/DNS configuration, etc.



For more information on telecommunications support, please visit the Telecommunications Support Office Home Page at the following address:

<http://vaww.va.gov/cso/>

SIGNON DELAYS

Users signing on to **VISTA** on a client workstation with the Broker Client Agent running should *not* experience any signon delays.

In order to provide users with the capability of single signon in both a GUI and roll-and-scroll Telnet session, the Kernel signon process was modified.

The Kernel signon process now tries to contact the RPC Broker V. 1.1 Client Agent on the client workstation (i.e., prior to and following the Access/Verify code prompts) to allow single signon to take place. A three-second (or less) delay is built into this process while attempting to connect to the Client Agent and allow for any possible network delays.

If you wish to eliminate the three second (or less) signon delay in a GUI/Telnet session (i.e., *not* associated with network delays), do either of the following:

1. Disable auto signon (a.k.a. single signon) for *all* users by setting the DEFAULT AUTO SIGN-ON field in the Kernel System Parameters file to "Disabled"
2. Install and run the Broker Client Agent on *all* client workstations, if auto signon is enabled on your system.



For more information on the DEFAULT AUTO SIGN-ON field, please refer to the "Integrated Single Signon for Multiple User Sessions" topic in Chapter 1 in this manual.

RPC BROKER FAQs



For examples of general or development-specific frequently asked questions (FAQs) about the RPC Broker, please refer to the following web site:

<http://vista2.med.va.gov/broker/>

Glossary

ACCESS CODE	A code that, along with the verify code, allows the computer to identify you as a user authorized to gain access to the computer. Your code is greater than six and less than twenty characters long; can be numeric, alphabetic, or a combination of both; and is usually assigned by a site manager or application coordinator. It is used by the Kernel's Sign-on/Security system to identify the user (see Verify Code).
ALERTS	Brief online notices that are issued to users as they complete a cycle through the menu system. Alerts are designed to provide interactive notification of pending computing activities, such as the need to reorder supplies or review a patient's clinical test results. Along with the alert message is an indication that the View Alerts common option should be chosen to take further action.
ANSI MUMPS	The MUMPS programming language is a standard recognized by the American National Standard Institute (ANSI). MUMPS stands for M assachusetts U tility M ulti- p rogramming S ystem and is abbreviated as M.
APPLICATION PACKAGE	Software and documentation that support the automation of a service, such as Laboratory or Pharmacy within VA medical centers. The Kernel application package is like an operating system relative to other VISTA applications.
CALLABLE ENTRY POINT	An authorized programmer call that may be used in any VISTA application package. The DBA maintains the list of DBIC-approved entry points.
CARET	A symbol expressed as up caret (^), left caret (<), or right caret (>). In many M systems, a right caret is used as a system prompt and an up caret as an exiting tool from an option. Also known as the up-arrow symbol or shift-6 key.
CLIENT	A single term used interchangeably to refer to the user, the workstation, and the portion of the program that runs on the workstation. In an object-oriented environment, a client is a member of a group that uses the services of an unrelated group. If the client is on a local area network (LAN), it can share resources with another computer (server).

COMPONENT	An object-oriented term used to describe the building blocks of GUI applications. A software object that contains data and code. A component may or may not be visible. These components interact with other components on a form to create the GUI user application interface.
COTS	Commercial Off-the-Shelf . COTS refers to software packages that can be purchased by the public and used in support of VISTA .
DATA DICTIONARY	<p>The Data Dictionary is a global containing a description of the kind of data that is stored in the global corresponding to a particular file. The data is used internally by VA FileMan for interpreting and processing files.</p> <p>A Data Dictionary (DD) contains the definitions of a file's elements (fields or data attributes), relationships to other files, and structure or design. Users generally review the definitions of a file's elements or data attributes; programmers review the definitions of a file's internal structure.</p>
DBIA	Database Integration Agreement , a formal understanding between two or more application packages which describes how data is shared or how packages interact. The DBA maintains a list of DBIAs between package developers, allowing the use of internal entry points or other package-specific features that are not available to the general programming public.
DEFAULT	A response the computer considers the most probable answer to the prompt being given. In the roll-and-scroll mode of VISTA , the default value is identified by double slash marks (//) immediately following it. In a GUI-based application the default may be a highlighted button or text. This allows you the option of accepting the default answer or entering your own answer. To accept the default you simply press the enter (or return) key. To change the default answer, type in your response.
DIRECT MODE UTILITY	A programmer call that is made when working in direct programmer mode. A direct mode utility is entered at the M prompt (e.g., > D ^XUP). Calls that are documented as direct mode utilities <i>cannot</i> be used in application package code.

DLL	<p>Dynamic Link Library. A DLL allows executable routines to be stored separately as files with a DLL extension. These routines are only loaded when a program calls for them. DLLs provide several advantages:</p>
	<ol style="list-style-type: none"> 1. DLLs help save on computer memory, since memory is only consumed when a DLL is loaded. They also save disk space. With static libraries, your application absorbs all the library code into your application so the size of your application is greater. Other applications using the same library will also carry this code around. With the DLL, you don't carry the code itself, you have a pointer to the common library. All applications using it will then share one image. 2. DLLs ease maintenance tasks. Because the DLL is a separate file, any modifications made to the DLL will not affect the operation of the calling program or any other DLL. 3. DLLs help avoid redundant routines. They provide generic functions that can be utilized by a variety of programs.
ERROR TRAP	<p>A mechanism to capture system errors and record facts about the computing context such as the local symbol table, last global reference, and routine in use. Operating systems provide tools such as the %ER utility. The Kernel provides a generic error trapping mechanism with use of the ^%ZTER global and ^XTER* routines. Errors can be trapped and, when possible, the user is returned to the menu system.</p>
FORUM	<p>The central e-mail system within VISTA. Developers use FORUM to communicate at a national level about programming and other issues. FORUM is located at the Washington, DC CIO Field Office (162-2).</p>
GUI	<p>Graphical User Interface. A type of display format that enables users to choose commands, initiate programs, and other options by selecting pictorial representations (icons) via a mouse or a keyboard.</p>
ICON	<p>A picture or symbol that graphically represents an object or a concept.</p>

IRM	Information Resource Management. A service at VA medical centers responsible for computer management and system security.
KERNEL	A set of M software routines that function as an intermediary between the host operating system and the VISTA application packages enabling packages to coexist in a standard OS-independent computing environment. The Kernel provides a standard and consistent user and programmer interface between application packages and the underlying M implementations.
MENU MANAGER	The Kernel module that controls the presentation of user activities such as menu choices or options. Information about each user's menu choices is stored in the Compiled Menu System, the ^XUTL global, for easy and efficient access.
MULTIPLE	A multiple-valued field; a subfile. In many respects, a multiple is structured like a file.
MUMPS (ANSI STANDARD)	A programming language recognized by the American National Standards Institute (ANSI) . The acronym MUMPS stands for Massachusetts General Hospital Utility Multi-programming System and is abbreviated as M.
NAMESPACING	A convention for naming VISTA package elements. The Database Administrator (DBA) assigns unique character strings for package developers to use in naming routines, options, and other package elements so that packages may coexist. The DBA also assigns a separate range of file numbers to each package.
NODE	In a tree structure, a point at which subordinate items of data originate. An M array element is characterized by a name and a unique subscript. Thus the terms: node, array element, and subscripted variable are synonymous. In a global array, each node might have specific fields or "pieces" reserved for data attributes such as name.
OPTION	As an item on a menu, an option provides an opportunity for users to select it, thereby invoking the associated computing activity. In VISTA , an entry in the OPTION file (#19). Options may also be scheduled to run in the background, non-interactively, by TaskMan.
PROMPT	The computer interacts with the user by issuing questions called <i>prompts</i> , to which the user returns a response.

REMOTE PROCEDURE CALL	A remote procedure call (RPC) is essentially M code that may take optional parameters to do some work and then return either a single value or an array back to the client application.
ROUTINE	A program or a sequence of instructions called by a program that may have some general or frequent use. M routines are groups of program lines that are saved, loaded, and called as a single unit via a specific name.
SECURITY KEY	The purpose of Security Keys is to set a layer of protection on the range of computing capabilities available with a particular software package. The availability of options is based on the level of system access granted to each user.
SERVER	The computer where the data and the Business Rules reside. It makes resources available to client workstations on the network. In VISTA , it is an entry in the OPTION file (#19). An automated mail protocol that is activated by sending a message to a server at another location with the "S.server" syntax. A server's activity is specified in the OPTION file (#19) and can be the running of a routine or the placement of data into a file.
SIGN-ON/SECURITY	The Kernel module that regulates access to the menu system. It performs a number of checks to determine whether access can be permitted at a particular time. A log of signons is maintained.
SUBSCRIPT	A symbol that is associated with the name of a set to identify a particular subset or element. In M, a numeric or string value that: is enclosed in parentheses, is appended to the name of a local or global variable, and identifies a specific node within an array.
UCI	User Class Identification, a computing area. The MGR UCI is typically the Manager's account, while VAH or ROU may be Production accounts.

USER ACCESS	<p>This term is used to refer to a limited level of access to a computer system that is sufficient for using/operating a package, but does not allow programming, modification to data dictionaries, or other operations that require programmer access. Any of VISTA's options can be locked with a security key (e.g., XUPROGMODE, which means that invoking that option requires programmer access).</p> <p>The user's access level determines the degree of computer use and the types of computer programs available. The Systems Manager assigns the user an access level.</p>
USER INTERFACE	<p>The way the package is presented to the user, such as Graphical User Interfaces that display option prompts, help messages, and menu choices. A standard user interface can be achieved by using Borland's Delphi Graphical User Interface to display the various menu option choices, commands, etc.</p>
VERIFY CODE	<p>The Kernel's Sign-on/Security system uses the verify code to validate the user's identity. This is an additional security precaution used in conjunction with the Access Code. Like the Access Code, it is also 6 to 20 characters in length. If entered incorrectly, it does not allow the user to access the computer. To protect the user, both codes are invisible on the terminal screen.</p>
VISTA	<p>Veterans Health Information Systems and Technology Architecture. VISTA includes the VA's application software (i.e., Windows-based and locally-developed applications, roll-and-scroll, and interfaces such as software links to commercial packages). In addition, it encompasses the VA's uses of new automated technology including the clinical workstations. VISTA encompasses the rich automated environment already present at local VA medical facilities.</p>
WINDOW	<p>An object on the screen that presents information such as a document or message.</p>

Index

- AUTO SIGN-ON, 1-15
- Broker
 - Listeners and Ports, 1-11
 - Message Structure, 1-18
- Changing the **VISTA** Verify Code, 2-6
- CLAGENT.EXE, 1-1, 1-15
- Client Agent, 1-1, 1-15
- Client Features, 1-1
- Client Manager, What Happened to it?, 1-9
- Client/Server Timeouts, 1-19
- Connect To Application Window, 1-2
- Connection Request
 - Validating, 2-1
- Customizing the Signon Window, 2-3
- DEFAULT AUTO SIGN-ON, 1-15
- DEFAULT MULTIPLE SIGN-ON, 1-15
- Domain Name Service (DNS), 1-3, 1-4, 1-5
- Edit Broker Servers Program, 1-2
- FAQs, 3-5
- Help files, 1-4
- HOSTS file, 1-3, 1-5, 1-14, 1-20, 3-3, 3-4
- Integrated Single Signon, 1-15
- KERNEL SYSTEM PARAMETERS file, 1-15
- LISTENERS
 - STARTING, 1-12
- Listeners and Ports, 1-11
- Menu Manager, 2-3
- Message Structure, 1-18
- MULTIPLE SIGN-ON, 1-15
- Network Connection, 3-4
- NEW PERSON file, 1-15
- Obtaining the Server TCP/IP Address, 1-11, 1-14
- OPTION file, 2-2
- OPTION SCHEDULING file, 1-13
- PING, 3-4
- Preface, iii
- Registry, 1-2, 1-10
- REMOTE PROCEDURE file (#8994), 1-18
- RPC Broker Diagnostic Program, How to test the Broker, 3-1
- RPCBI.DLL, 1-9
- RPCs
 - Validating, 2-7
- RPCTEST.EXE, 3-1, 3-2
- Schedule/Unschedule Options, 1-13
- Security, 2-1
 - Change **VISTA** Verify Code
 - Component, 2-6
 - Features, 2-1
 - Sample Security Procedures, 2-8
 - Signon Window, Customizing, 2-3
 - Summary of Tasks, 2-8
 - Validating Connection Request, 2-1
 - Validating RPCs, 2-7
 - Validating Users, 2-1
- Server Features, 1-11
- ServerList.EXE, 1-2
- Servers, using the Edit Broker Servers Program, 1-2
- Signon Window, Customizing, 2-3
- Signon Window, sample, 2-2
- Single Signon, 1-15
- STARTING LISTENERS, 1-12
- STOP^XWBTCP(Listener port), 1-12
- STRT^XWBTCP(Listener port, 1-12
- System
 - Features, 1-1
 - Overview, 4
- Table of Contents, v
- Test the Broker Using the RPC Broker Diagnostic Program, 3-1
- Timeouts, 1-19
- Troubleshooting, 3-1
- Validating

Index

Connection Request, Security, 2-1	Verify Code, Changing, 2-6
RPCs, Security, 2-7	VISTA.INI file, What Happened to it?, 1-9
Users, Security, 2-1	Windows Registry, 1-2, 1-10
Verify and Test the Network Connection, 3-4	WINSOCK.DLL, 1-18, 3-4